

UDC: 004.7

INFO M: str. 21-27

UPRAVLJANJE ŽIVOTNIM CIKLUSOM APLIKACIJA KORIŠĆENJEM TEAM FOUNDATION SERVER PLATFORME APPLICATION LIFECYCLE MANAGEMENT USING TEAM FOUNDATION SERVER PLATFORM

Ognjen Pantelić, Mišljen Filip
Fakultet organizacionih nauka Univerziteta u Beogradu

REZIME: U oblasti razvoja softvera primetan je trend pojave sve većih i zahtevnijih softverskih projekata za čiji je razvoj potrebno angažovati dosta osoba (programeri, testeri, menadžeri itd.) i utrošiti dosta radnih sati. Zato se došlo do potrebe za korišćenjem određenih alata prilikom razvoja aplikacija koje objedinjuju potrebe svih učesnika u razvoju proizvoda i koji povećavaju efikasnost i produktivnost prilikom razvoja softverskog projekta. Na tržištu postoji velikih broj rešenja koja imaju potrebne funkcionalnosti, kao što je i Microsoft Team Foundation Server (TFS). U radu će biti prikazana analiza tog alata i da se prikažu arhitektura i funkcionalnosti Team Foundation Server platforme tj. alata. Praktični deo rada se bavi razvojem aplikacije na ovoj platformi prilikom čijeg razvoja će se iskoristiti neke od funkcionalnosti koje nudi TFS.

KLJUČNE REČI: Team Foundation Server (TFS), projekat, razvoj softvera

ABSTRACT: In field of software development there is notable trend of developing bigger and more demanding software products for whose development there has to be engaged many people (programmers, testers, managers, etc.) and spend many working hours. That's why there is a need to use some tools while developing software that incorporate the needs of all participants in software product and increase efficiency and productivity of development of software product. There are many solutions that have some of the above mentioned functionalities, and among them is Microsoft Team Foundation Server (TFS). The goal of master thesis is to analyze and present architecture and functionalities of Team Foundation Server platform. In practical part of these thesis application will be developed on these platform for whose development some of the TFS functionalities will be used.

KEY WORDS: Team Foundation Server (TFS), project, software development.

1. UVOD

Alati za upravljanje životnim ciklusom projekata služe da omoguće efikasnije planiranje, razvoj i održavanje softverskih projekata. Generalno, ovakvi alati omogućavaju efikasnije upravljanje projektima. Potreba za takvim alatima se javila razvojem velikih softverskih projekata. Na razvoju i održavanju tih velikih projekata može da učestvuje dosta timova čiji članovi mogu biti iz različitih struka: programeri, menadžeri projekata, rukovodioci, testeri, poslovni analitičari itd. Kako bi ti timovi efikasnije saradivali prilikom izrade softverskih rešenja potrebno je da se koriste neki alati i mehanizmi za njihovu efikasniju saradnju i kordinisanje rada prilikom razvoja i održavanja softverskih rešenja.

Na tržištu postoji velikih broj rešenja koja imaju te funkcionalnosti, a među njima jedan je i Microsoft Team Foundation Server (TFS) kojim će se baviti ovaj rad. Rad će se baviti analizom alata TFS. Biće predstavljena njegova arhitektura i njegove funkcionalnosti. Glavna funkcionalnost koja će se analizirati i na kojoj je fokus ovog rada je kontrola verzije fajlova. Sa ovom funkcionalnosti su indirektno povezane sve ostale funkcionalnosti ovog alata. Biće predstavljene funkcionalnosti za upravljanje projektima tj. radnim zadacima. Predstavice se i neke napredne funkcionalnosti alata: Izveštavanje o stanju projekta, automatizacija izrade gotovih softverskih rešenja, proširljivost TFS-a, upravljanje privilegijama i rad sa TFS-om iz web čitača.

U praktičnom delu rada će se prikazati korišćenje funkcionalnosti ovog alata prilikom razvoja aplikacije za vođenje podsistema nabavke jedne firme. U TFS-u će se izvršiti plani-

ranje razvoja ove aplikacije, razviciće se aplikacija korišćenjem funkcionalnosti koje nudi ovaj alat i omogućiće se pravljenje izvršnog softverskog rešenja ove aplikacije.

2. MICROSOFT TEAM FOUNDATION SERVER

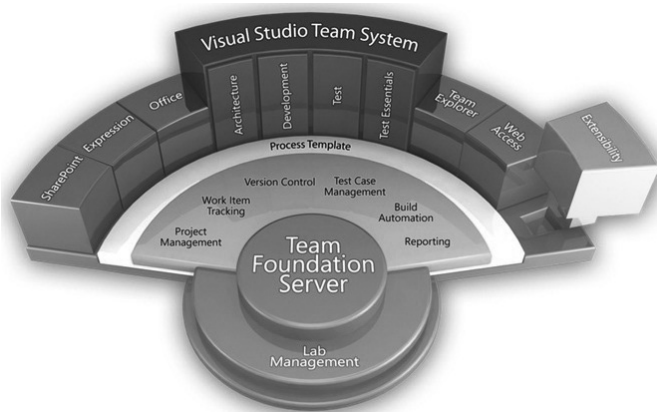
Vendori se već dugu niz godina bave pravljenjem zahtevnih softverskih rešenja. Prilikom pravljenja tih softverskih rešenja javila se potreba da se efikasno kontroliše paralelni rad na istom projektu, prate defekti (eng. bugs), prate radni zadaci koji su dodeljeni osobama koje učestvuju u projektu, upravlja izradom gotovog softverskog rešenja itd. [1] TFS je sveobuhvatan alat za saradnju i upravljanje projektima izgrađen u .NET platformi. Omogućava upravljanje izvornim kodom, omogućava korisnicima da stvaraju i upravljaju delovima posla tj. radnim zadacima (eng. work item), automatizuje proces izgradnje (eng. build) aplikacije i druge funkcionalnosti. [1] TFS je softverska platforma namenjena za efikasniju saradnju između timova i njihovih članova. TFS je softver usmeren ka optimizaciji procesa timskog razvoja softvera. [1] Platforma je softversko i/ili hardversko okruženje u kome se izvršava program. Za TFS se kaže da je platforma jer se razvoj aplikacija odvija u okviru njegovog repozitorijuma.

Arhitektura po kojoj je realizovan TFS je klijent-server arhitektura. Pošto se serverski sloj deli na dva podsloja, smatra se da je TFS realizovan na osnovu troslojne arhitekture, slika 1:

1. Klijentski sloj
2. Serverski sloj
 - a. Aplikativni sloj
 - b. Sloj podataka [1]

Arhitektura TFS-a je po mestu čuvanja podataka centralizovana. Drugim rečima, postoji samo jedan glavni sloj podataka tj. jedna glavna baza podataka gde se čuvaju svi podaci nastali na osnovu rada na TFS-u. Sve osobe koje žele da učestvuju u razvoju projekta moraju da preuzimaju podatke i da ih čuvaju u toj bazi podataka [4]. Osnovna instalacija TFS-a obuhvata instalaciju ovog alata samo na serverskom sloju, jer je potrebno posebno instalirati serverski deo TFS-a na server, a klijenti moraju sami na svojim računarima da instaliraju klijentsku aplikaciju preko koje ostvaruju komunikaciju sa serverskim slojem TFS-a i pozivaju funkcionalnosti koje se izvršavaju na serverskom sloju.

2.1 Funkcionalnosti



Slika 1. - TFS funkcionalnosti, izvor [2]

Glavne funkcionalnosti koje nudi TFS su, slika 1:

- Kontrola verzije (eng. version control) – Služi za upravljanje fajlovima koji zahtevaju verzioniranje tj. praćenje izmena u fajlovima i za obezbeđivanje mehanizama za paralelni rad na projektu.
- Praćenje radnih zadataka (eng. work item tracking) – Služi za praćenje zahteva, zadataka, defekata, scenarija itd. Radni zadaci su osnova za planiranje i praćenje stanja projekta.
- Funkcije projektnog menadžmenta (eng. project management functions) – Omogućava planiranje timskog projekta zasnovanog na korisnički specifikovanim softverskim procesima koji omogućavaju planiranje i praćenje razvoja i održavanja softverskog projekta korišćenjem Microsoft Office Excel i Project alata..
- Kreiranje izvršnih softverskih rešenja (eng. build)
- Izveštavanje (eng reporting) – Predstavlja pomoć u proceni stanja i napretka projekta.
- Team Project Portal – Predstavljen je kao Microsoft Windows SharePoint servis koji pruža mesto komunikacije za timski projekat.
- Team Foundation Shared Services koji pružaju brojne infrastrukturne servise koji su nevidljivi za krajnjeg korisnika.
- Proširljivost – TFS omogućava da programeri razvijaju svoje aplikacije koje mogu da koriste postojeće funkcionalnosti TFS-a i da proširuju njegove postojeće mogućnosti [1].

2.2 Radni zadaci

TFS nije namenjen samo programerima tj. članovima tima koji razvijaju i doraduju kod softverskog projekta. TFS poseduje i funkcionalnosti koje su namenjene projekt menadžerima. Te funkcionalnosti omogućavaju upravljanje projektima tj. definisanje radnih zadataka, praćenje preostalog rada na projektu, praćenje aktivnih defekata (eng. bug), praćenje raspoloživosti programera, komunikacija sa programerima, pravljenje izveštaja itd. [3].

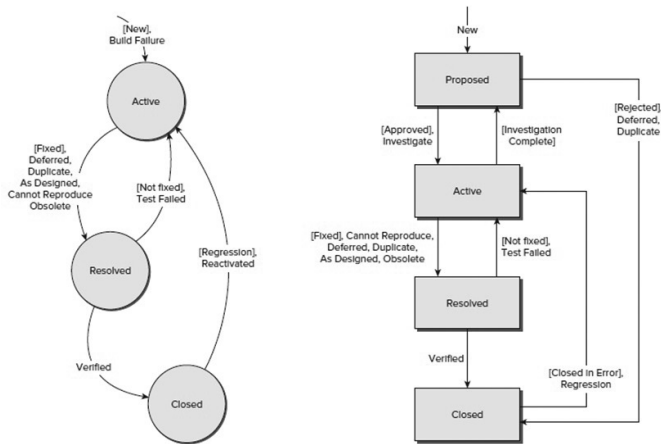
Definisanje radnog zadatka je osnovna funkcionalnost projektnog menadžmenta u TFS-u. Microsoft definiše radni zadatak kao zapis u bazi podataka koji se u TFS-u koristi za praćenje napredovanja projekta. Radni zadaci se najčešće koriste prilikom planiranja razvoja projekta [3].

Za radni zadatak je moguće definisati neke atribute počevši od naslova, tipa, oblasti, iteracija i raznih drugih. Postoji više tipova radnih zadataka: Defekat, Problem, Zadatak, Test slučaj, Korisnički zahtev i Ostali tipovi. Pregled radnog zadatka na TFS-u se može videti na slici 2.

Slika2. - Unos i pregled radnog zadatka na TFS-u

Radni zadaci mogu biti međusobno povezani tj. biti u nekoj vezi jedan sa drugim (Dete – roditelj, Test odnos, Prethodnik/sledbenik,Povezani,Ostale veze).Povezivanje radnih zadataka proširuje mogućnosti TFS-a prilikom izveštavanja i upravljanja projektom. Npr. u Microsoft Office Project-u se mogu najbolje vizuelno videti ove veze između radnih zadataka. TFS ne poseduje mehanizam provere da li su veze između radnih zadataka semantički ispravne. Npr. moguće je definisati kružne veze između radnih zadataka [3].

Kad se radni zadatak doda na TFS najčešće mu je dozvoljeno da bude u tačno jednom određenom statusu. Npr. kada se kreira radni zadatak najčešće mu je status predložen tj. nije moguće uneti ga na TFS tako da mu je status završen iako i jeste završen [3]. Slično kao i kod fajlova na TFS-u, moguće je videti kompletnu istoriju promene nekog dela radnog zadatka tj. koja su se polja u okviru radnog zadatka menjala i kada. Moguće je vizuelno predstaviti promene statusa radnog zadatka gde se vidi ko je i kada menjao status radnog zadatka. Npr. kada se doda defekat na TFS, onda je on u stanju aktivan. Programer mu menja stanje u rešeno kada ga ispravi. Ako tester potvrdi da više ne može da reprodukuje defekat onda mu menja stanje u zatvoren, u suprotnom mu vraća stanje u aktivno, slika 3.



Slika 3. - Moguća stanja zadataka u agilnim i MSF metodologijama, prema [4]

Grupi fajlova koji su izmenjeni i čije izmene trebaju da se potvrde na TFS-u se može dodeliti jedan ili više radnih zadataka tj. može se definisati veza između izmenjenih fajlova i radnih zadataka. Ovako se mogu lakše ispratiti promene u kodu jer se za izmenjene fajlove mogu videti radni zadaci za koje se vezani i u kojima su detaljnije opisani zahtevi koje je potrebno ispuniti. Prilikom dodeljivanja radnog zadatka nekoj izmeni na TFS-u, može se podesiti da je radni zadatak ovom izmenom završen (eng. resolved) ili se može podesiti da postoji neka veza između radnog zadatka i izmene (eng. associate with) [3].

Radni zadaci ne mogu da se obrišu iz sistema preko Team Explorera jer organizacije koje koriste TFS ne žele da se važni radni zadaci slučajno ili namerno obrišu. Brisanje je dozvoljeno samo preko konzolne aplikacije TFS-a korišćenjem odgovarajuće komande [3]. Radni zadaci se na TFS-u mogu pronalaziti preko upita. Npr. na TFS-u je moguće da postoji veliki broj zadataka zbog velikog broja projekata koji se razvijaju na njemu. Upiti omogućavaju lakše pronalaženje tačno određenih zadataka tako što se definišu vrednosti koja polja u okviru radnih zadataka moraju da zadovolje. Npr. moguće je napraviti upit kojim se prikaže lista zadataka koje je planirano da uradi osoba koja je trenutno radi na TFS-u. U TFS-u je moguće definisati šablon rada (eng. process template) prilikom rada na radnim zadacima. Šablon rada predstavlja implementaciju metodologije rada na projektu u TFS. Šablonom se definiše način rada i upravljanja projektom. Za svaki projekat na TFS-u je potrebno podesiti šablon rada. Šabloni omogućavaju definisanje mogućih statusa radnih zadataka, dozvoljene vrednosti polja u radnim zadacima, moguće veze između radnih zadataka itd. Npr. Microsoft Solutions Framework (MSF) metodologija za razliku od agilne metodologije dozvoljava da stanje radnog zadatka bude predloženo kao na slici 4. TFS poseduje definisane MSF i skram šablone. Moguće je razviti nove ili preuzeti postojeće sa neke lokacije i dodati ih u TFS [3].

2.3. Build projekata

Build je druga najvažnija funkcionalnost (posle kontrole verzije) koja omogućava poboljšanje kvaliteta razvoja aplikacija [3]. Pravljenje izvršnih fajlova projekta se naziva build projekta. Npr. kod desktop aplikacija build može da služiti

za pravljenje fajla za instalaciju te aplikacije (najčešće sa nazivom setup.exe). Kod web aplikacija zadatak builda je najčešće postavljanje fajlova web aplikacije na web server kome korisnici pristupaju kako bi radili sa aplikacijom.

Build projekta se sastoji od:

1. Kompajliranje izvornog koda u izvršni kod.
2. Pakovanje izvršnog koda u izvršne fajlove kao što su MSI, ClickOnce, JAR fajlovi.
3. Izvršavanje testova - Prilikom builda projekta mogu se pokrenuti automatizovani testovi koji proveravaju da li aplikacija radi kako treba tj. da li daje očekivane rezultate za svaki test. Ukoliko se testovi uspešno izvrše i vrate željene rezultate onda to znači da izmene u aplikaciji od prethodnog builda najverovatnije nisu izazvale neke probleme u funkcionisanju nekih delova aplikacije na koje se testovi odnose.
4. Postavljanje izvršnih fajlova na okruženje (npr. kopiranje instalacionih fajlova na određenu lokaciju ili kopiranje web fajlova na server).
5. Kreiranje build dokumentacije tj. Izveštaja [3].

Sklapanje jednog po jednog dela aplikacije i kreiranje izvršnih fajlova na osnovu njih je najčešće kompleksan i dugotrajan proces sklon greškama. Zato postoji automatizovani build, koji kad se iskonfiguriše kako treba, omogućava kreiranje izvršnih fajlova projekta putem samo jedne akcije. Build je funkcionalnost koju poseduje Visual Studio razvojno okruženje i većina drugih razvojnih okruženja. TFS proširuje mogućnosti builda putem njegove automatizacije [3].

Automatizovani build projekta se izvršava na serveru, a ne na klijentskom računaru, jer ukoliko se projekat uspešno builduje na serveru na kome projekat nije razvijan, onda je velika verovatnoća da neće doći do problema prilikom pokretanja aplikacije kod korisnika [3]. Kada aplikativni sloj TFS-a primi zahtev za izvršavanje builda ili kad sam kreira zahtev za izvršavanje automatizovanog builda, zahtev se prosleđuje build kontroleru. Build kontroler je deo aplikativnog sloja koji služi da odluči kom serveru će proslediti taj zahtev na izvršavanje. Serveri koji mogu da izvršavaju build se nazivaju build agenti. Build agent predstavlja ili običan server ili virtuelnu mašinu na serveru koja se aktivira prilikom prijema zahteva za buildom i na kojoj se izvršava build. Kontroler može biti samo jedan dok agenata može biti više. Ovime se postiže bolja skalabilnost u zavisnosti od učestalosti slanja zahteva za izvršavanje builda i od računarskih resursa koje zahteva build neke aplikacije. Kontroler i jedan agent mogu fizički biti i na istom serveru, ali se to ne preporučuje jer je pravljenje builda procesorski i memorijski veoma zahtevna operacija pa bi ukoliko se build izvršava na istoj mašini gde je i aplikacioni sloj, to moglo da utiče na performanse aplikativnog sloja što može dosta da utiče na produktivnost rada. Svaki build se na kraju čuva na posebnoj lokaciji (eng. drop location) tj. na toj lokaciji se čuva arhiva prethodno izvršenih buildova. Sve informacije u vezi builda (redni broj builda, da li je uspešno izvršen i ostali podaci o izvršavanju) se čuvaju u bazi podataka iz koje je moguće naknadno prikazati podatke o prethodnim izvršavanjima builda [3].

Ukoliko je broj zahtevanih buildova veći od broja agenata onda se definicije onih buildova koji ne mogu odmah da se pokrenu čuvaju u redu u kome su pristigli kod build kontrolera.

Nakon što neki build agent završi sa izvršavanjem builda, obavestava se kontroler da je taj agent slobodan i onda kontroler prosleđuje neki build iz reda (u zavisnosti od prioriteta i vremena pristizanja u red) tom agentu. Različiti build kontroleri mogu da šalju zahteve za izvršavanje buildova na zajedničke build agente [3]. Build se nakon izvršavanja (bez obzira da li je uspešno ili neuspešno izvršen) čuva na određenoj lokaciji (eng. drop location). Ukoliko je neuspešno izvršen onda se na toj lokaciji čuva fajl sa informacijama o greškama koje su se javile prilikom builda. Nakon nekog vremena ukupna veličina memorijskog prostora koju zauzimaju buildovi može dosta da poraste. Zato je potrebno podesiti koliko će se poslednjih buildova čuvati u memoriji build servera. Ovo se podešava za svaku definiciju builda i za svako stanje builda posebno.

3. RAZVOJ APLIKACIJE NA TFS-U

Sa obzirom na veličinu koju bi zauzela puna dokumentacija aplikacije (slučajevi korišćenja, sistemski dijagrami sekvenci, ugovori o sistemskim operacijama, ekranske forme itd.) zbog broja slučajeva korišćenja i da akcenat rada nije na predstavljanju te dokumentacije, u praktičnom delu rada će biti dati osnovni referentni delovi te dokumentacije kako bi se pokazalo planiranje i realizacija aplikacije korišćenjem Team Foundation Server alata. Arhitektura aplikacije data je na slici 4.

3.1 Korisnički zahtev - verbalni opis

Potrebno je projektovati i imlementirati informacioni sistem sektora nabavke. U ovom sektoru se posmatraju sledeći procesi koje je potrebno implementirati u informacioni sistem:

- Slanje zahteva za ponudu dobavljačima
- Prijem ponuda
- Slanje porudžbine ka dobavljaču
- Prijem naručene robe

Arhitektura softverskog sistema je izvedena iz troslojne arhitekture i sastavljena je iz sledećih nivoa:

1. Korisnički interfejs (C# Windows Forms)
2. Aplikaciona logika (C#)
3. Skladište podataka (Microsoft SQL Server baza podataka)

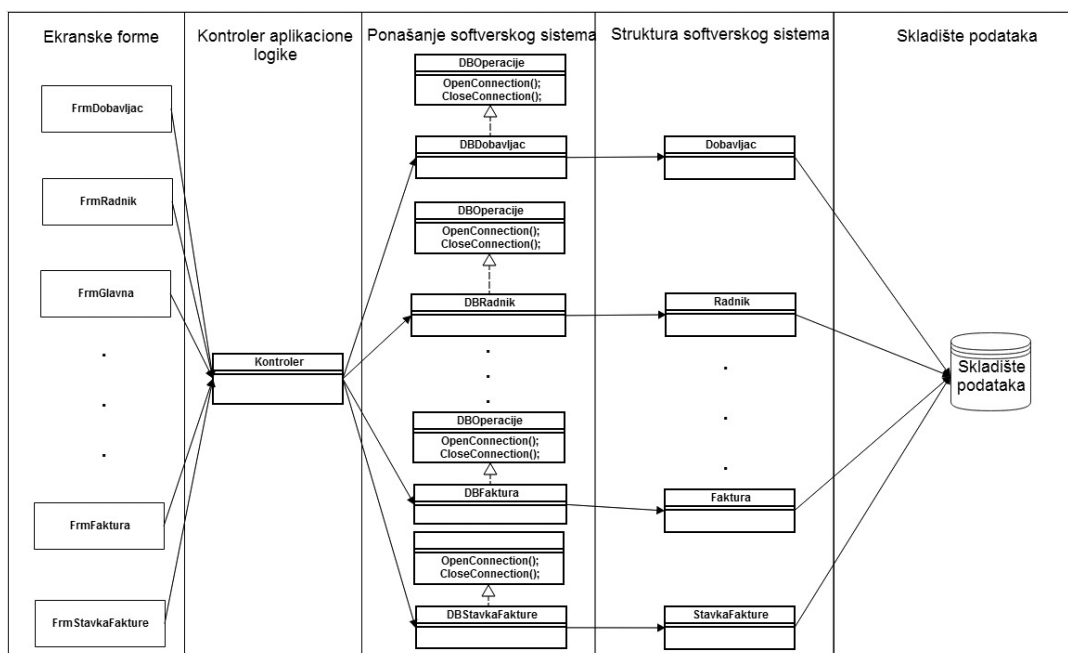
Preko klase Kontroler prihvataju se zahtevi od klijenata za izvršenje sistemskih operacija koji se zatim prosleđuju do odgovarajućih klasa koje su odgovorne za izvršenje sistemskih operacija. Tj. klasa kontroler predstavlja posrednika između frontend i backend dela aplikacije. Na slici 6 je prikazana arhitektura sistema.

3.2 Planiranje u TFS-u

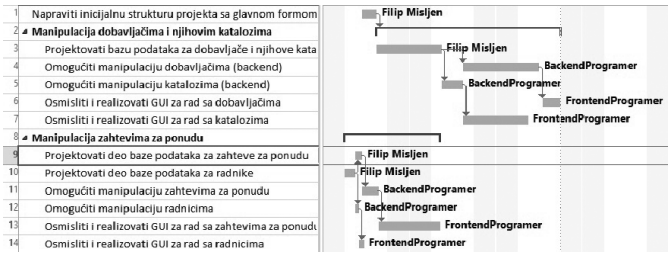
U TFS-u je izvršeno planiranje projekta preko funkcionalnosti upravljanja radnim zadacima. Definiše se radni zadatak koji se sastoji od više manjih radnih zadataka. Na slici 5 su prikazani primeri veza između radnih zadataka. Na slici 6 je dat pregled liste radnih zadataka u Microsoft Office Project programu i gantogram tih radnih zadataka.

ID	Work It...	Title
Changeset (1 items)		
Changeset 38		
Child (5 items)		
16	Task	Projektovati bazu podataka za dobavljače i njihove kataloge
17	Task	Omogućiti manipulaciju dobavljačima (backend)
18	Task	Omogućiti manipulaciju katalogizima (backend)
19	Task	Osmisliti i realizovati GUI za rad sa dobavljačima
20	Task	Osmisliti i realizovati GUI za rad sa katalogizima
Predecessor (1 items)		
14	Task	Napraviti inicijalnu strukturu projekta sa glavnom formom v1.0.0
Successor (1 items)		
21	Task	Manipulacija zahtevima za ponudu v1.0.2

Slika 5. - Veze između radnih zadataka



Slika 4. - Arhitektura aplikacije



Slika 6. - Gantogram u Microsoft Office Project-u

Pre početka pisanja koda i izrade same aplikacije, iskorišćen je TFS-ov objektni model kako bi se postavila određena pravila koja moraju da budu zadovoljena prilikom potvrđivanja izmena fajlova. Dodato je pravilo koje onemogućava potvrđivanje izmena ukoliko je komentar:

1. Isti kao i naziv trenutnog korisnika. Često korisnici TFS-a unose za komentar njihove inicijale ili imena kako bi brzo potvrdili izmenu, ali je tako teže njima samima nakon nekog vremena da identifikuju šta je tom izmenom urađeno kao i drugim korisnicima.
2. Kraći od 10 karaktera. Na ovaj način se korisnik primorava da malo duže opiše izmenu.

U slučaju da oba pravila nisu zadovoljena ispisaće se obe poruke o grešci. Kod je napisan u C# projektu koji je kompajliran u fajl sa ekstenzijom dll. TFS je podešen da učitava i sva pravila koja su u tom fajlu.

Programski kod pravila:

```
using System;
using System.Collections.Generic;
using System.Text;
using Microsoft.TeamFoundation.VersionControl.Client;
using Microsoft.VisualStudio;

namespace CheckInPravila
{
    [Serializable]
    public class Policies : PolicyBase
    {
        public Policies()
        {
        }

        public override string Description
        {
            get
            {
                return "Komentar mora da sadrži bar 10 karaktera"+
                    " i da ne bude različit od naziva trenutnog Windows korisnika.";
            }
        }

        public override bool Edit(IPolicyEditArgs policyEditArgs)
        {
            return true;
        }

        public override PolicyFailure[] Evaluate()
        {
            List<PolicyFailure> narusenaPravila = new List<PolicyFailure>();
            string komentar = PendingCheckin.PendingChanges.Comment;

            string korisnik = Environment.UserName;

            if (komentar.Trim().ToLower().Equals(korisnik.ToLower()))
            {
                PolicyFailure narusenoPravilo = new PolicyFailure(
                    String.Format("Komentar ne može da bude isti kao i"
                        + " naziv trenutnog Windows korisnika: {0}.", korisnik), this);
                narusenaPravila.Add(narusenoPravilo);
            }
        }
    }
}
```

```
}
if (komentar.Trim().Length < 10)
{
    PolicyFailure narusenoPravilo = new PolicyFailure(
        "Komentar mora da sadrži najmanje 10 karaktera.", this);
    narusenaPravila.Add(narusenoPravilo);
}
return narusenaPravila.ToArray();
}

public override string Type
{
    get { return "Pravilo za duzinu i naziv komentara"; }
}

public override string TypeDescription
{
    get
    {
        return "Komentar mora da sadrži najmanje 10 karaktera"
            + " i da ne bude različit od naziva trenutnog Windows"
            + " korisnika. U slučaju da su sva pravila narušena"
            + " ispisaće se sve poruke o narušenim pravilima";
    }
}
```

Na kraju se dobija upozorenje prilikom potvrđivanja izveštaja.

3.3 Implementacija

Projekat je realizovan u Microsoft Visual Studio 2012 razvojnom okruženju u C Sharp programskom jeziku. Korišćene su funkcionalnosti besplatnog Team Foundation Server 2012 Express izdanja. Zbog korišćenja ove verzije TFS-a nije bilo moguće kreiranje izveštaja i prikaz funkcionalnosti integrisanja TFS-a sa SharePoint-om. Ceo projekat je realizovan na jednom računaru te su i serverski (sloj podataka i aplikativni sloj) i klijentski sloj instalirani i konfigurisani da budu na istom računaru. Zbog toga je prilikom instalacije odabrano jedino moguće rešenje što se tiče build arhitekture tj. postoji samo jedan build agent i on se zajedno sa build kontrolerom nalazi na istom računaru.

Kako bi se u nekoj meri simulirao višekorisnički rad na TFS-u, na istom računaru su kreirana tri Windows korisnika kojima su dodata odgovarajuća prava na TFS-u:

1. Filip Misljen - Administratorski korisnik. Zadužen za razvoj osnovne strukture projekta i razvoj baze podataka.
2. BackendProgramer - Zadužen za razvoj backend dela aplikacije.
3. FrontendProgramer - Zadužen za razvoj frontend dela aplikacije.

Baza podataka je realizovana u Microsoft SQL Server 2012 sistemu za upravljanje bazama podataka.

Sa obzirom da se aplikacija stalno razvija tj. izdaju se nove verzije aplikacije, potrebno je na neki način obezbediti da korisnici pređu sa starije na noviju verziju aplikacije. Pored korišćenje novije verzije aplikacije, potrebno je obezbediti da klijenti pređu na noviju verziju baze podataka. Kreiran je jedan fajl koji sadrži SQL skriptove za kreiranje baze podataka. Kako se model baze podataka proširuje tako se na kraj ovog tekstualnog fajla dodaju novi skriptovi (za kreiranje tabela, kolona, uskladištenih procedura itd.). Nakon što se završi iteracija samo se kreira nova verzija tog fajla koja je identična aktuelnoj sa tom razlikom što se postavi komentar koji objašnjava da je to zvanična verzija baze.

```

Server: S:\MasterRad\PodsistemNabavke\Baza\SriptZaBazu.txt;C33
145
146 INSERT INTO [dbo].[Verzija] VALUES ('1.0.0')
147
Local: C:\Users\Filip\Desktop\PodsistemNabavke\PodsistemNabavke\Baza\SriptZaBazu.txt
145
146 INSERT INTO [dbo].[Verzija] VALUES ('1.0.0')
147
148 USE [ProjekatBaze]
149 GO
150
151 /***** Object: Table [dbo].[Adresa]    Script Date: 20/1
152 IF EXISTS (SELECT * FROM sys.objects WHERE object_id = 0E
153 DROP TABLE [dbo].[Adresa]
154 GO
155
156 /***** Object: Table [dbo].[Adresa]    Script Date: 20/1
157 SET ANSI_NULLS ON
158 GO
159
160 SET QUOTED_IDENTIFIER ON
161 GO
162
163 IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id =
164 BEGIN
165 CREATE TABLE [dbo].[Adresa](
166     [adresaID] [int] IDENTITY(1,1) NOT NULL,
167     [mesto] [nvarchar](50) NULL,
168     [ulica] [nvarchar](50) NULL,
169     [brojzgrade] [smallint] NULL,
170     CONSTRAINT [PK_Adresa] PRIMARY KEY CLUSTERED
171     (
172         [adresaID] ASC
173     )WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNC
174     ) ON [PRIMARY]
175 END
176 GO
177
178
179
180 USE [ProjekatBaze]
181 GO
182
183 IF EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id
184 ALTER TABLE [dbo].[Dobavljači] DROP CONSTRAINT [FK_F

```

Slika 7. - Prikaz razlike između dve verzije baze

U fajlu SkriptZaBazu.txt se razvijala baza podataka. Zvanične verzije baze su 1.0.0 i 1.0.1. Ukoliko neki klijent želi da dobije aktuelnu verziju softverskog projekta 1.0.1, a poseduje verziju 1.0.0 onda je potrebno samo da pusti skriptove koje dobije kada uporedi razliku između te dve verzije, slika 8. Za čuvanje verzije baze podataka u istoj bazi podataka je dodata tabela Verzija sa kolonom DBVersion u kojoj se čuva aktuelna verzija baze kod klijenta i ta se verzija prikazuje u aplikaciji.

3.4 Referentni slučajevi korišćenja

U ovom radu biće izdvojena samo dva slučaja korišćenja.

SKI: Pregled stavki kataloga dobavljača

Naziv SK: Pregled stavki kataloga dobavljača

Akter SK: Korisnik aplikacije (u daljem tekstu radnik)

Učesnici SK: Radnik i program (u daljem tekstu sistem)

Preduslov: Sistem je uključen. Otvorena je forma Dobavljači. Učitani su svi dobavljači.

Osnovni scenario SK:

1. Radnik bira dobavljača čije kataloge želi da prikaže (APUSO)



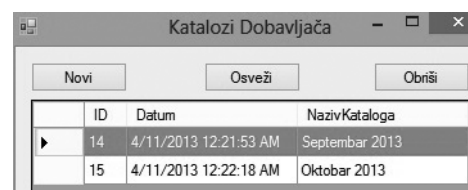
Slika 8. - Odabir dobavljača

2. Radnik poziva sistem da prikaže odabrane kataloge (APSO)



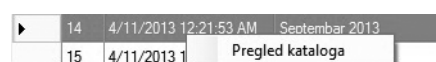
Slika 9. - Prikaz odabranih kataloga

3. Sistem pronalaži odabrane kataloge (SO)
4. Sistem prikazuje podatke o katalozima (IA)



Slika 10. - Prikaz podataka o katalozima

5. Radnik bira katalog čije stavke želi da prikaže (APUSO)
6. Radnik poziva sistem da prikaže odabrane stavke (APSO)



Slika 11. - Prikaz odabranih stavki kataloga

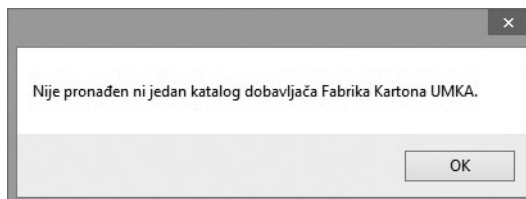
7. Sistem pronalaži odabrane stavke (SO)
8. Sistem prikazuje podatke o stavkama (IA)

Masina/Lokacija	Cena	Proizvod
Pokretna traka	560	Elektricni prekidac
Pokretna traka	300.5	Elektricni prikljucak

Slika 12. Prikaz podataka o stavkama

Alternativni scenario:

Ako sistem nije uspeo da pronađe ni jedan katalog trenutnog dobavljača, sistem prikazuje poruku o tome (IA) i preki da izvršenje scenaria.



Slika 13. - Obaveštenje da katalog nije pronađen

4. ZAKLJUČAK

Kako su ulaganja u IT projekte izuzetno velika, postavljaju se prioriteta i odabir projekata koji će se finansirati jedan je od bitnih aspekata upravljanja IT investicijama. Ekonomska opravdanost ulaganja sredstava, odnosno efekti koji se postižu uvođenjem jednog informatičkog rešenja u poslovanju su nekada teško merljivi [5]. U današnje vreme potrebni su alati koji bi integrisali različite aktivnosti i direktno uticali opravdanost IT investicije. Jedan od takvih alata je i TFS. Danas je izuzetno teško razviti „ozbiljan“ i veliki softver bez upotrebe nekog alata za upravljanje verzijom fajlova i zajednički rad na projektima. Prelazak na korišćenje nekog alata za kontrolu verzije fajlova je pozitivan korak u razvoju projekta u odnosu na razvoj proizvoda bez takvog alata.

TFS poseduje sve funkcionalnosti koje su potrebne za razvoj bilo kakvih projekata i omogućava integrisanje različitih aktivnosti koje je potrebno izvršiti u toku razvoja softvera. Omogućava visok stepen automatizacije onih aktivnosti kod kojih je moguća automatizacija rada. Korišćenjem TFS-a se obezbeđuje kontinuirani proces koji obezbeđuje saradnju menadžmenta i softverskih inženjera na svim poljima delovanja, od sakupljanja zahteva i arhitekture aplikacija, preko programiranja, testiranja i praćenja razvoja projekta, pa sve do isporuke i održavanja softverskog sistema. TFS je moguće integrisati sa Sharepoint alatom za razmenu podataka između ra-

zličityh interesnih stana u okviru projekata. Preko Microsoft SQL Server baze podataka je realizovan repozitorijum TFS-a. U Microsoft SQL Server Reporting servisima je moguće pravi razne vrste izveštaja o stanju i napretku projekta. TFS verzija 2012 omogućava korišćenje Git repozitorijuma. Preostalo je da se dorade i neki nedostaci u okviru TFS-a koji bi korisnicima omogućili efikasniji rad kao što je na primer direktna pretraga fajlova u repozitorijumu po nazivu ili nekom drugom kriterijumu. Prilikom odabira alata za kontrolu verzije ili za upravljanje životnim ciklusom projekata jedan od kriterijuma koji bi trebalo da bude značajan za odlučivanje je integracija tog alata sa razvojnim alatom u kome se razvija projekat. TFS se u potpunosti integriše sa Visual Studio okruženjem i omogućava direktan jednostavan i brz pristup svim funkcionalnostima koje TFS nudi.

Jedan od problema koji je uočen pri korišćenju TFS-a je podešavanje automatizovanog builda. Bilo bi poželjno da se podešavanje automatizovanog builda malo više približi novim korisnicima TFS-a. Jednom kada je automatizovani build uspešno konfigurisan i izvršen kako treba više se nisu javljali problem u toku njegovog izvršavanja.

5. LITERATURA

- [1] Microsoft Corporation, Dennis Minium, „Team Foundation Server Fundamentals: A Look at the Capabilities and Architecture“ [http://msdn.microsoft.com/en-us/library/ms364062\(v=vs.80\).aspx](http://msdn.microsoft.com/en-us/library/ms364062(v=vs.80).aspx) (poslednja poseta 31.10.2013.)
- [2] Ardent Infotech, „Team Foundation Server Development“ http://ardentinfotech.com/services/tfs_development/tfs_development.html (poslednja poseta 31.10.2013.)
- [3] Ed B., Martin W., Grant H., Brian K., „Professional Team Foundation Server 2012“, pp 112-409, 2013.
- [4] Jason Y., „Introduction to Distributed Source Control“ <http://www.ytechie.com/> (poslednja poseta 31.10.2013.)
- [5] Ana P., Measuring performance of it investments, INFOM vol. 10, br. 40, UDC 004:658.152, 2011



Ognjen Pantelić docent, Fakultet organizacionih nauka Univerziteta u Beogradu

Kontakt: pantelic.ognjen@fon.bg.ac.rs

Oblasti interesovanja: ERP sistemi, IT menadžment, IS u sportu i saobraćaju, integrisana softverska rešenja, logičko projektovanje baza podataka



Mišljen Filip, MSc information systems, - Fakultet Organizacionih nauka

Kontakt: filipmisljen@hotmail.com

Oblasti interesovanja: Razvoj softvera, poslovna inteligencija, baze podataka, web tehnologije

